K Sreram

Software Engineering · Machine Learning · Module Designing · Object oriented design and development

Chennai, Tamil Nadu, India

(+91) 94-4478-4785 | sreramk26@gmail.com | http://www.intelligentdream.com/ | sreramk | k-sreram-a04a90b7

"Understanding the fundamental working of Intelligence and incorporating them into technology"

Education

Easwari Engineering College

Chennai, Tamil Nadu, Inida

B.S. in Computer Science and Engineering

2014 - PRESENT

Chettinad Vidyashram RA Puram

Chennai, Tamil Nadu, India

High School (11th and 12th)

2012 - 2014

- Created an image editing application in TurboC. It converts a 24 bit bitmap image into a 4 bit image, and displays the image with BGI graphics

Skills

Designing object oriented solutions to problems.

proficient in both design and development

- All my projects, which are, MapDB, Graph, Neural Network library, the Sudoku puzzle solver, the inverted indexer, the conversational engine, etc, are structured using the object oriented paradigm and uses all the object oriented design and development features.
- I always give extra preference to code readability and documentation.

Languages: C, C++, Java and Python (with Cython).

I've experience using the above languages

- I have used C in the past to write embedded programs, using Arduino.
- Most of my projects use C++ as their core language
- Learning a new language (preferably object oriented) or starting to use a new library will be an easy task for me. I can start using a language almost instantly and become good at it in a week.

Decoding functionality of 3rd-party code, from documentation and source code.

I am good at reading and understanding 3rd party code.

- I had to read and understand the boost-python library once, which only had the official documentation and had no tutorials (except for the official ones).
- I read the MongoDB's official documentation (C++ driver) to understand and build the MapDB library.

Re-factoring, optimizing and scaling code.

proficient at managing the readability vs scalability trade-off.

- All my projects are scaled for high performance and are readable at the same time

Designing novel solutions to problems.

I honour novelty above analogy

- The Sudoku puzzle solver contains a novel method to reduce iterations and improve solving time.
- I have designed and implemented all the projects listed in the document from the scratch.
- Therefore I'm good at handling new and uncertain situations.

Explaining information in a comprehensible manner

I write articles in various domains

- I maintain a blog, containing more than 60 articles.
- **URL:** `http://www.intelligentdream.com/`

Writing machine learning solutions

I come up with novel ways to improve accuracy

- I have solved few regression problems (supervised).
- I have designed and implemented unsupervised solutions to solve small games.
- I use TensorFlow with python.

Projects

A Conversational Engine

(Privately maintained project)

A general purposed conversational engine as a marketing tool. *March 2018 - PRESENT*

- **Brief:** This conversational engine learns to converse in place of a person in a conversation by observing the replies given to each statement.
- **Language(s) used:** C++14.
- **Use case:**This module attempts to takeover the place of the marketer or salesman by listening to their conversation pattern and successfully marketing the same product/service to future customers
- **Measuring success:** Must provide highly dynamic interface that helps amateurs developers / casual users to easily extend the functionality of the conversational engine at will.

Indexer-Note and the Indexer

(privately maintained project)

An Indexer-Node and an Indexer that is used by the N-Gram algorithm. *Feb. 2018 - Mar. 2018*

- **brief :** An indexer (which can be used for any arbitrary string-based indexing) that tries to index nodes in a complex graph.
- **Language(s) used:** C++14.
- **dependencies :** N-Gram algorithm, MapDB

N-Gram algorithm

(Privately maintained project)

An N-Gram algorithm that stores data in the database (uses MapDB) *Jan. 2018 - Feb. 2018*

- **brief :** Stores association between words (both preceding and succeeding). Thus enabling us to roughly estimate the probability of the occurrence of a word preceding a particular word and succeeding a particular word.
- **Language(s) used:** C++14.
- **dependencies :** MapDB, Graph-connector
- **usage :** Reconstructing sentences based on frequency of occurrence of a particular word following another word.

Graph-Connector

(Privately maintained project)

A Graph Connector that uses the MapDB and structured connections in the hard-disk *Jan. 2018 - Feb. 2018*

- **brief :** Uses the MapDB system to maintain doubly indexed connections to other objects (referred to by an unique identification number) that contains a word-based index and a weight based index
- **Language(s) used:** C++14.
- **dependencies :** MapDB
- **usage :** This can be used to form connections between multiple nodes that reside in the hard-disk. Both word-based query (accepting a string) and weight-based queries can be invoked.

MapDB - storing highly structured Key Value data in the database.

(Privately maintained project)

A MongoDB wrapper abstraction, for key-value pair storage *Dec. 2017 - Jan. 2018*

- **Brief:** Creates std::multimap like abstraction, that helps store data of a wide range of custom C++ types (defined as a structure) associated either with a string or with a 64-bit intiger value.
- **Language(s) used:** C++14.
- It also has methods that lets us Iterate through the storage like we do using std::multimap. But with a major difference that the records are stored in the hard-disk rather than the RAM.
- **Status:** The first version was successfully implemented, and works as expected
- **Measuring success:** In this project's context, success is defined as creating a class that can be used like std::miltimap, providing all its functionality which includes having its key records sorted.
- **Future work / Improvement:** As of now, the key supports only string values and signed 64-bit integers. We can improve it to support binary records as keys, allowing it to hold any arbitrary record as the key. But we will have to include a custome compression operator for this to work
- **Limitations:** Fundamentally this MapDB system uses MongoDB. Therefore, by default, it inherits all the limitations present in MongoDB database system.
- A single record cannot be longer than 16 MB size.
- The size of the record's key must always be less than or equal to 1024 bytes - (MongoDB's structural overhead) - (MapDB's structural overhead)

AI Kernel library

(Privately maintained project)

A library that implements a game-loop- Registers methods and maintains an environment. *Nov. 2017 - Dec. 2017*

- **Brief:** Implements an environment that has a shared storage base, and every registered module/method will have access to this storage base.
- **Language(s) used:** C++14.
- **Basic Working:** The registered modules runs in an infinite loop, until the 'terminate' hook is activated.
- **Further Explanation:** The system contains hooks, which registered modules, and each activated hook gets invoked in an infinite look. For each invocation of the hook, the registered methods are invoked one after the other. All registered methods can manage the environment
- **Storage Environment:** Each method can define a unique local scope and always has access to the global scope. There can be methods without any local scope. Every method in the system will have access to the storage-base.
- **Measuring Progress/success:** This project can be considered successful if it can be used to create systems that contain ensemble of rules, and if a decisive result can be incurred from the collaborative functioning of the ensemble.
- **Limitation:** Extensively uses macros, to ease the development process. As a result, this system becomes unportable, making it difficult to compile it as a dynamic library and still use all the features in-built within the library.

Key-Value pair database

(Privately maintained project)

A library that implements a key-value pair database. *Nov. 2017 - Dec. 2017*

- **Brief:** Implements an easy to use, extensible, random-access record based Key-Value pair database system. The key should be of a fixed size (assigned by the user, for each database); the record is segmented and stored, enabling random access read and write.
- **Language(s) used:** C++11.
- **Basic Working:** This database system uses balanced binary-search tree for structuring the data in the hard-disk (Instead of B+ trees)
- **Reason for using Binary Search tree instead of B+ tree:** This can support constructing database recursively (i.e., one database inside another). This is done by making the lowest level api calls of a particular database instance to use the highest level api calls of a different instance.
- **Measuring Progress/success:** The lowest level api-calls must be made virtual. Once this is done, the project will be "complete". I measure success by ensuring that the database occupies the least amount of space and if we can recursively construct databases.
- **Limitation:** Though using binary search tees bring with it the advantage of creating recursive database systems, the number of read/write accesses will be slightly higher when we use balances binary search trees.

Neural Network Library

(publicly maintained - Open Source)

A learning toolkit for neural-network, written using Java *Oct. 2016 - Nov. 2016*

- **URL:** `https://github.com/Aplokodika/MachineLearning/tree/master/src`
- **Brief:** A generic neural network implementation which makes it easy to design any arbitrary ANN structure; I have implemented the back-propagation algorithm for the learning part.
- **Language(s) used:** Java 8.
- A graph based implementation. Each node in the graph literally represents a neuron.
- Helps explore the solution space by plotting a graph that depicts the minima and maxima region in a 2D graph. Any two weight values can be chosen for the plot.
- **limitations:** Does not use GUPs or parallel processing. Because this implementation represents Neurons as nodes in a graph (literally), the computation process consumes a bit more time than the systems that are implemented as a series of matrix manipulation.
- **Measuring success:** Must help in exploring the error space of the neural-network system as the training iteration proceeds. As of now, this project does satisfy this case. But newer features must be added to make this observation more comprehensive

Sudoku puzzle solver

(publicly maintained - Open Source)

A Sudoku puzzle solve that works for 9x9 puzzles. *2015*

- **URL:** `https://github.com/sreramk/Sudoku-`
- **Brief:** Sudoku puzzle solver that uses a rule-bases approach. It switches over to brute-force if there are multiple solutions to the problem.
- **Language(s) used:** C++.
- I have implemented a method that tries to minimize the number of iterations in the brute-force search by choosing the squares (while filling the squares using back-tracking brute-force) that ensure that the board reveals most information on the final solution.
- A 2.5GHs machine will solve almost any 9x9 puzzle less than a second.
- **limitations:** Does not use parallel computing. The code is not easily extensible to accept arbitrary dimension problems. For example, this implementation cannot be easily modified to support 3x3, 6x6 27x27 or other types of puzzles.
- **Measuring success:** Factors that define success in the context of the Sudoku Puzzle solver are, solving the problem in the least number of iterations possible, consuming the least amount of time.